A Simulation Based Approach to Solve A Specific Type of Chance Constrained Optimization

Lijian Chen
Department of MIS, Operations Management, and Decision Sciences,
University of Dayton[1]

Abstract: We solve the chance constrained optimization with convex feasible set through approximating the chance constraint by another convex smooth function. The approximation is based on the numerical properties of the Bernstein polynomial that is capable of effectively controlling the approximation error for both function value and gradient. Thus we adopt a first-order algorithm to reach a satisfactory solution which is expected to be optimal. When the explicit expression of joint distribution is not available, we then use Monte Carlo approach to numerically evaluate the chance constraint to obtain an optimal solution by probability. Numerical results for known problem instances are presented.

Keywords: Chance constrained optimization, Monte Carlo, Convex optimization

1. Introduction

In health care, supply chain management, and many other industries, the decision makers demand a technique to guarantee the service level under uncertainties. For example, hospitals need to ensure that 95% or more patients are seen by a doctor within a short period of time; power planners require power transmission systems to function reliably with almost zero chance of blackout; suppliers need sufficient network capacity to ensure the successful transportation of goods with a chance of over 90%. A deterministic mathematical programming problem is

$$\min\{f_0(x) : f_i(x) \leq 0, i = 1, \ldots, m, x \in \mathbb{R}^n\}.$$

Let I denote a set of indices i such that the constraint $f_i(x) \leq 0$ was contaminated with random vector $\xi : \Omega \to \mathbb{R}^s$ of a probability space $(\Omega, \mathcal{B}, \mathbb{P})$. These constraints $f_i(x, \xi) \leq 0, i \in I$, as a whole, represent an event which demands serious attention of the decision makers. This event was desirable but not required almost surely. Thus, we impose a probabilistic measure

$$\mathbb{P}\{f_i(x, \xi) \leq 0, i \in I\} \geq 1 - \alpha$$

[1]300 College Park Ave, Dayton, OH 45469, Tel.: +1-937-229-2757, lchen1@udayton.edu

to represent the fact that decision makers may need this desired event by the chance of $1-\alpha$, at least. The model becomes

$$\min\{f_0(x) : f_j(x) \leq 0, j \notin I, \mathbb{P}\{f_i(x,\xi) \leq 0, i \in I\} \geq 1-\alpha\}.$$

Chance-constrained optimization has little progress made until recently due to its complexity. It may be the case that the only way to check the feasibility of a given point is to use Monte Carlo simulation. To calculate the joint cumulative distribution function explicitly, we also need considerable investment on computers. In addition, the feasible set can be non-convex even if $f_i(x,\xi), i \in I$ are linear with respect to x. Thus, the research on the chance-constrained optimization has gone into two somewhat different directions. One is to transform the original problem into a combinatorial problem such as [6] through discretizing the probability distribution. Another is to approximate the chance constraint by a convex function to extract function value and gradient. The obtained solutions are mostly satisfactory but sub-optimal. These solutions seem to be very practical and reasonable for certain problem instances. However, the heuristics solution may raise the concern of the solution quality due to lack of either functional or probabilistic linkage to the true optimal solution.

In this paper, we only investigate a specific type of chance-constrained optimization in which the random vector appears in the right-hand side of the affine mapping and the distribution of ξ is jointly *continuous*, and log-concave. The reason that we impose these conditions is to equivalently transform model (1) to a convex optimization. This reason is based on the following results from the literature.

Theorem 1. *If $\xi \in \mathbb{R}^m$ is a random vector that has log-concave probability distribution, then the function*

$$F_\xi(Dx) := \mathbb{P}\{Dx \geq \xi\}$$

is log-concave with respect to x. $D \in \mathbb{R}^{m\times n}$ is a real matrix of coefficients. Moreover, the set

$$\{x \in \mathbb{R}^n : \mathbb{P}(Dx \geq \xi) \geq 1-\alpha\}$$

is convex and closed.

The proof is in [11, Chapter 5].

In business analytics, the decision is usually subject to multiple resource constraints. We impose a joint probability measure on a handful of constraints to ensure a $1-\alpha$ chance of realization. We have

$$\begin{aligned} &\min\ f_0(x) \\ \text{Subject to: } &f_j(x) \leq 0, j \notin I && \text{(Resource constraints)} \\ &\mathbb{P}(Dx \geq \xi) \geq 1-\alpha,\ x \in \mathbb{R}^n && \text{(Chance constraint)} \end{aligned}$$

where the distribution of ξ is joint log-concave and continuous and D is deterministic. Therefore, the above has a convex and closed feasible set. Let

$$g(x) := \log(1-\alpha) - \log(F_\xi(Dx)) \leq 0$$

where F_ξ is the cumulative distribution function of the random vector ξ. $g(x)$ is a convex function with respect to $x \in \mathbb{R}^n$ and the chance-constrained optimization model is equivalent to

$$\begin{aligned} \min\ & f_0(x) \\ \text{subject to: } & g(x) \leq 0 \\ & f_i(x) \leq 0, i \notin I \end{aligned} \tag{CCCF}$$

If other constraints, $f_0(x), f_i(x), i \notin I$, are convex, model (CCCF) is a convex optimization problem. The term CCCF stands for a Chance Constrained problem with Convex Feasible set. We are not endorsing the superiority of convex modeling but emphasizing the fact that the convex optimization will always yield global and reliable solutions. Moreover, the convex optimization with smooth constraints will always terminate within a polynomial number of iterations without acquiring the second-order information (see [8, Chapter 2]).

The key contribution of this research is to approximate the functional value, $g(x)$, and gradient, $\nabla g(x)$, by a Bernstein polynomial-based function. First, the Bernstein polynomial can effectively control the approximation errors of function values and gradients *simultaneously*. Second, the degree of polynomial can be well-controlled by Jackson's theorem V (see [1]). Third, our approximation will preserve the convexity and the differentiability of $g(x)$. The log-concave assumption will be less likely to become a serious obstacle of implementation because many commonly-used distributions are indeed continuous and log-concave. For instance, uniform distribution, normal distribution, beta distribution, Dirichlet distribution, Pareto distribution, and gamma distribution (when the shape parameter is greater than one) are continuous and log-concave.

We need to show another primary reason that we adopt the Bernstein polynomial in this research. One may argue that there are many routes to approximate such a chance constraint and we agree. We have no intention to endorse our approach over another approximation scheme developed in parallel. However, given the fact that $g(x)$ is convex, our approximation is mathematically guaranteed to be convex and, at the same time, our approximation will stay within an error range uniformly. In particular, we need not only the functional value but also the gradient. It is always a good idea to approximate a convex function with another convex function when both function and its first-order information are extracted. In comparison to a non-convex approximation to the original function, a convex approximation has no local zigzag fluctuations which may considerably affect the approximation of the first-order information such as gradient or sub-gradient. A guaranteed convex function will ensure very limited complication when evaluating the gradient.

The remainder of this paper is organized as follows. The approximation approach is described in Section 2, beginning with reasoning and details followed by the approximation procedure. In Section 3, we describe our approach for the chance-constrained optimization along with the computational complexity. In Section 4, we discuss the impact of using Monte Carlo to evaluate the cumulative distribution function. We also show that the obtained optimal solution, if Monte

Carlo is adopted, will converge to the original optimal in probability. In Section 5, we present the spreadsheet-based implementation which is a combination of business analytic and convex optimization solvers. We conclude this paper in Section 6 with concluding remarks about this new approach.

2. A polynomial approximation approach

In order to highlight the approximation approach, we further relax the non-linear smooth but convex constraints with linear functions.

$$\begin{aligned} &\min\ c'x \\ &\text{subject to: } g(x) \leq 0 \\ &\qquad\qquad Ax \leq b, x \in \mathbb{R}^n \end{aligned} \tag{CCCFL}$$

where c represents the coefficients for the objective function. The new set of constraints, $Ax \leq b$, is in the place of the resource constraints $f_j(x) \leq 0, i \notin I$. We name this problem the Chance Constrained optimization with Convex Feasible set and Linear resource constraints (CCCFL).

The function $g(x)$ is defined on $\mathbb{R}^n$ onto $\mathbb{R}$. Thus, the value of $g(x)$ will be a scalar and the gradient $\nabla g(x)$ is a n-dimensional vector. The i^{th} component is $\dfrac{\partial g(x)}{\partial x^i}$, $x := [x^1; \ldots; x^n]$ where x^i represents the ith component of x, i.e. $\nabla g(x) = \left[\dfrac{\partial g(x)}{\partial x^1}; \ldots; \dfrac{\partial g(x)}{\partial x^n}\right]$. Let us define the i^{th} marginal function of $g(x)$ as $g_i(x^i) : [a^i, b^i] \to \mathbb{R}$ with respect to $x^i \in [a^i, b^i]$ and other component $x^j, j \neq i$ fixed. We would evaluate $\dfrac{\partial g(x)}{\partial x^i}$ by calculating the derivative of the marginal function $g_i(x^i)$ that $\dfrac{\partial g(x)}{\partial x^i} = \nabla g_i(x^i)$. Since $g(x)$ is convex, all of its marginal functions $g_i(x^i), i = 1, \ldots, n$ are convex with respect to x^i. Our approach is to approximate all the marginal functions of $g_i(x^i)$ with convex, differentiable polynomial of degree k, $q_k(x^i)$ at a fixed x. We use the vector $[q'_k(x^1); \ldots; q'_k(x^n)]$ to estimate $\nabla g(x)$.

We show the construction of $q_k(x^i)$ to approximate the marginal function $g_i(x^i) : [a^i, b^i] \to \mathbb{R}$. This idea is inspired by the Weierstrass theorem that any continuous function can be approximated by a polynomial with adequately large number of order. However, the choice of proper polynomial will considerably affect the approximation performance. In this research, we adopt a polynomial named Bernstein polynomial:

Definition 1. *The Bernstein polynomial of function $\phi(y) \in C^1[0,1]$ is*

$$B_k(\phi; y) = \sum_{j=0}^{k} \binom{k}{j} y^j (1-y)^{k-j} \phi(j/k)$$

for any $k = 0, 1, 2, \ldots$.

For the sake of simplifying notations, we use $\phi(y)$ in the place of $g_i(x^i)$. Without loss of generality, we assume that $\phi(y)$ is on $[0,1]$ as we can always scale a 1-1 mapping from a closed interval $[a^i, b^i]$ to $[0,1]$. There is a theorem associated with the Bernstein polynomial.

Theorem 2 (Voronovskaja theorem). *If $\phi(y)$ is bounded on $[0,1]$, differentiable in some neighborhood of y and has second derivative $\phi''(y)$ for some $y \in [0,1]$, then*

$$\lim_{k\to\infty} k|\phi(y) - B_k(\phi; y)| = \frac{y(1-y)}{2}\phi''(y).$$

If $\phi(y) \in C^2[0,1]$, the convergence is uniform.

The proof of this theorem is in many papers such as [9]. This theorem states that the value of $B_k(\phi; y) - \phi(y)$ tends to zero at the speed of $\frac{1}{k}$ where k represents the degree of the approximating Bernstein polynomial. That is, any smooth function would be approximated by Bernstein polynomial of degree k with arbitrary accuracy as $k \to \infty$. Nevertheless, the approximation using Bernstein polynomial is rather poor because, to halve the error, we have to increase the degree from k to $2k$. In order to improve the approximation performance, we can use Bernstein polynomial to approximate the second order derivate of the original function which is based on the following result.

Theorem 3. *There exists a sequence of component functions,*

$$\psi_0(y), \psi_1(y), \psi_2(y) \ldots, \tag{1}$$

each convex on $[a,b]$, such that any convex function $\phi(y) \in C^1[a,b]$ may be approximated with arbitrary accuracy on $[a,b]$ by a sum of non-negative multiples of the component functions.

This theorem is extremely important to our research, and we present it here as a courtesy from the original source.

Proof: It will be adequate to show this conclusion on $[0,1]$ because we can make a linear change of variable, if necessary, to transform any finite interval $[a,b]$ onto $[0,1]$. Let us suppose that we wish to approximate to a given convex function $\phi(y) \in C^1[0,1]$. At first, by Votonovskaja theorem, it is valid to assume that $\phi(y)$ is continuously twice differentiable. We use the Bernstein polynomials indirectly and write

$$B_k(\phi''; y) = \sum_{j=0}^{k} \binom{k}{j} y^j (1-y)^{k-j} \phi''(j/k) \tag{2}$$

Let us observe that $y^j(1-y)^{k-j} \geq 0$ on $[0,1]$ and that in (2) $\phi''(y)$ is being approximated by a sum of non-negative multiples of the polynomial $y^j(1-y)^{k-j}$.

For $k \geq 2$, define $q_k(y)$ by

$$q_k''(y) = B_{k-2}(\phi''; y);\ q_k'(0) = \phi'(0);\ q_k(0) = \phi(0) \tag{3}$$

We see that $q_k(y)$ is a polynomial of degree at most k. We also define $\beta_{j,k}(y)$, for $2 \leq j \leq k$, by

$$\beta''_{j,k}(y) = y^{j-2}(1-y)^{k-j};\ \beta'_{j,k}(0) = \beta_{j,k}(0) = 0 \tag{4}$$

To complete the definition of the polynomials $\beta_{j,k}(y)$, we define,

$$\beta_{0,k}(y) = sign[\phi(0)];\ \beta_{1,k}(y) = y \times sign[\phi'(0)] \tag{5}$$

The relevance of the choice of function (5) will be seen later. We then have that

$$q_k(y) = \sum_{j=0}^{k} c_j \beta_{j,k}(y) \tag{6}$$

$c_j \geq 0$ and $\beta''_{j,k}(y) \geq 0$ on $[0,1]$. Now, given any $\epsilon > 0$, there exists an integer k for which

$$|B_{k-2}(\phi''; y) - \phi''(y)| < \epsilon,\ |q''_k(y) - \phi''(y)| < \epsilon \tag{7}$$

on $[0,1]$ and therefore, for $y \in [0,1]$,

$$\left| \int_0^y (q''_k(t) - \phi''(t))dt \right| \leq \int_0^y |q''_k(t) - \phi''(t)|dt \leq \epsilon y \leq \epsilon \tag{8}$$

Using (3), the inequality (8) give

$$|q'_k(y) - \phi'(y)| < \epsilon \text{ and } |q_k(y) - \phi(y)| < \epsilon \tag{9}$$

Recalling the definition of $q_k(y)$ in (6), we see that this last inequality (9) completes the proof for case when $\phi''(y)$ exists. Note that the polynomial $\beta_{j,k}(y)$ may be enumerated as $\psi_0(y), \psi_1(y), \psi_2(y), \ldots$. □

We highlight key results here. First, the inequality (9) shows that our approach is capable of controlling the approximation errors of $\phi(y)$ and $\phi'(y)$ *simultaneously*. In comparison to our approach, a general polynomial approximation by Weierstrass theorem can only approximate arbitrarily close to the function rather than both function and gradient at the same time. Second, we can always numerically consider any convex function on $[0,1]$ to be continuously twice differentiable. This property is very important to our research that we can *always* assume high-order differentiability of $g(x)$. Otherwise, we can always construct a higher-order Bernstein polynomial which approximate $g(x)$ to within $\dfrac{\epsilon}{2}$ on $[0,1]$ to ensure (9).

We now address the choice of the non-negative coefficients c_j. From the previous analysis, we set

$$\begin{aligned} &\psi_j(y) = \beta_{j,k}(y), 2 \leq j \leq k \\ &\psi_0(y) = sign[\phi(0)] \\ &\psi_1(y) = y \times sign[\phi'(0)] \end{aligned} \tag{10}$$

and the polynomials $\beta_{j,k}(y)$ are easily computed because for $j \geq 2$,

$$\beta''_{j,k}(y) = y^{j-2}(1-y)^{k-j} = y^{j-2}\sum_{\ell=0}^{k-j}(-1)^{\ell}\binom{k-j}{\ell}y^{\ell}. \tag{11}$$

Thus,

$$\beta_{j,k}(y) = y^{j}\sum_{\ell=0}^{k-j}\frac{(-1)^{\ell}\binom{k-j}{\ell}y^{\ell}}{[(\ell+j)(\ell+j-1)]} \tag{12}$$

We construct

$$q_k(y) = \sum_{j=0}^{k} c_j\psi_j(y).$$

The value of c_j are determined

$$\min_{c}\left\{\max_{s=1,\ldots,k+1}\{\phi(y_s) - \sum_{j=0}^{k} c_j\psi_j(y_s)\} :\ c_j \geq 0\right\} \tag{13}$$

In the above model, we take $k+1$ observations on $[0,1]$, $(y_s, \phi(y_s)), s = 1, \ldots, k+1$ where y_s are predetermined values on $[0,1]$. This model is a convex optimization problem with $k+1$ variables and its computational complexity is $O((k+1)^3)$ at the worst case (see [3]). The approximation constructed by (13) is named the *best approximation* or *least square approximation* (see [4]).

In addition, we still need to determine the choices on the degree of k and the corresponding $k+1$ points, i.e., $(y_s, \phi(y_s)), s = 1, \ldots, k+1$. By theorem 3, the approximation error can be controlled by increasing k. However, in practice, high-degree polynomials usually cause Runge's phenomenon (see [9]), which means that the approximation is extremely unstable. The primary cause of Runge's phenomenon is due to the polynomial of high degree such as a degree of 1,000 or more. The following theorem addresses the issue that the Runge's phenomenon is less of a concern to our approach because we only need a low-degree polynomial.

Theorem 4 (Jackson's theorem). *If $\phi(y)$ is r-differentiable on $y \in [0,1]$ and $\phi(y)$ is approximated by the best approximation $q_k(y)$ of degree k (constructed by theorem (13)), then the approximate error of $\phi(y)$ on $[0,1]$ by $q_k(y)$ satisfies,*

$$\max_{y\in[0,1]}|\phi(y) - q_k(y)| \leq \left(\frac{\pi}{2}\right)^{r}\frac{|\phi^{r}(\omega)|}{[(k-r+2)\ldots(k)(k+1)]}, k \geq r$$

where $\phi^{r}(\omega)$ represents the r-order derivative of $\phi(y)$ at some $\omega \in [0,1]$ and $y_s, s = 1, \ldots, k+1$ are distinct predetermined points on $[0,1]$.

This is called Jackson's Theorem V introduced in [4, Page 147]. The derivation of the theorem is very lengthy and technical. We now present the history and the outline of this theorem and its impact to our approach. In 1911, D. Jackson presents the first Jackson's theorem as follows:

Theorem 5. *Given $f(x)$ is a continuous function with a period 2π, has r-th derivative ($r > 1$), and satisfies the inequality*

$$|f^{(r)}(x)| \leq 1.$$

Then, there exists for any positive integer n a trigonometric sum *$P_{n-1}(x)$ of order less than n and this trigonometric sum satisfies the inequality*

$$|f(x) - P_{n-1}(x)| \leq \frac{A_r}{n^r}, 0 \leq x \leq 2\pi,$$

and hence the value of A_r depends solely on the value of r.

This theorem is later published in [7]. Essentially, this theorem states that the approximate error is bounded and the reduction on error is extremely fast as we increase the degree of the approximate polynomial. Next, every even trigonometric polynomial can be expressed as an algebraic polynomial and conversely. Hence, the error in the best approximation by *trigonometric polynomial* is the same as the error in the best approximation by *algebraic polynomials*.

Jackson's Theorem V is extremely important result supporting our approach. In theory, it shows that if we approximate a r-differentiable function by algebraic polynomials, the error will be fast reduced by increasing the order of polynomial. Thus, Runge's phenomenon is less of a concern because the approximate polynomial of a very limited degree would be adequate to control the approximation error. We need to remark that the assumption of r-differentiable $\phi(y)$ can be justified by the Voronovskaja theorem. Although we find the coverage of the differentiability of probability functions in [12, Section 4.4.1], we need to justify the r-differentiability with $r > 1$. The argument is that if $\phi(y)$ is indeed r- differentiable, we can apply the theorem directly. Otherwise, we need to apply the Voronovskaja theorem as long as the original function is convex. We can always find a smooth approximation constructed by Bernstein polynomial which is r-differentiable. Moreover, the value of $\phi^r(\omega)$ is well bounded in a close and bounded interval with a predetermined r because of the similar argument suggested by the Voronovskaja theorem.

We need to adopt the Chebyshev nodes on $[0, 1]$ to select $k + 1$ distinct points $(y_s, \phi(y_s)), s = 1, \ldots, k + 1$ to control the error, $\max_{y\in[0,1]} |\phi(y) - q_k(y)|$. Although Jackson's Theorem guarantees to bound the error, a good selection of $(y_s, \phi(y_s)), s = 1, \ldots, k+1$ will greatly reduce the error (see [13, Lecture 20]) in practice. When we interpolating $\phi(y)$ at point $(y_s, \phi(y_s)), s = 1, \ldots, k + 1$, the error is

$$\max_{y\in[0,1]} |\phi(y) - q_k(y)| = \Pi_{s=1}^{k+1}(y - y_s)\frac{\phi^k(\omega)}{(k + 1)!} \tag{14}$$

The term $\Pi_{s=1}^{k+1}(y-y_s)$ needs to be minimized and thus we adopt the Chebyshev nodes. The *Chebyshev nodes* on $[0, 1]$ which are described as follows.

$$y_s := \frac{1}{2} - \frac{1}{2}\cos\left(\frac{2s - 1}{2k + 2}\pi\right), s = 1, \ldots, k + 1$$

When Chebyshev nodes are implemented, $\Pi_{s=1}^{k+1}(y-y_s)$ will be minimized to $\frac{1}{2^k}$ and if we assume that $|\phi^k(y)| \leq M$, we have

$$\max_{y\in[0,1]} |\phi(y) - q_k(y)| \leq \frac{1}{2^k(k+1)!}M. \tag{15}$$

At a given error bound $\epsilon > 0$ for $g(x)$, we need to control the error of every $\phi(y)$ no greater than $\frac{\epsilon}{n}$. Thus, the necessary degree of $q_k(y)$ should be

$$\min \left\{ k : \frac{1}{2^k(k+1)!}M < \frac{\epsilon}{n} \right\} \tag{16}$$

Increasing the degree of $q_k(y)$ from k to $k+1$ means a reduction of approximation error by $2(k+1)$, which is extremely fast. As k keeps growing, the reduction of error will be much faster. Thus, it is extremely unlikely that our approximation approach will require high-degree polynomials. For example, $n = 20, M = 100, \epsilon = 10^{-4}$, we present the values of $\frac{n}{2^k(k+1)!}M$ by distinct degrees in table 1. We also take the error bound of degree $k = 5$ as the baseline and present the relative precentage of error with greater degrees to demonstrate the fast reduction in the approximate error. With such a fast rate of error reduction, our approach will not suffer the Runge's phenomenon at all.

k values	$k=5$	$k=6$	$k=7$	$\mathbf{k=8}$	$k=9$	$k=10$
$\frac{n}{2^k(k+1)!}M$	0.0868	0.0062	3.87×10^{-4}	$\mathbf{2.15\times10^{-5}}$	1.07×10^{-6}	4.89×10^{-8}
Error reduction	100%	7.1429%	0.4459%	0.0248%	0.0012$	0.0001%

Table 1: Extremely fast convergence by degree of approximating polynomial

There are many approximate approaches being developed in parallel. We have neither intention nor interest to show the superiority against all the approximation approaches. However, our approach does outperform the approximation of some other polynomial of degree k. For example, $1, y, y^2, \ldots, y^k$, the approximation function may fail to preserve convexity and the performance of evaluating gradient is poor. Let $q_k(y)$ denote the approximation function of our approach and $\bar{q}_k(y)$ represent the approximation function constructed from $1, y, y^2, \ldots, y^k$ rather than polynomial (10). In Table 2, when $y = 0, y = 0.2, y = 0.7$ and $y = 1.0$, the gradients of our approximation functions considerably outperform their counterparts constructed from $1, y, y^2, \ldots, y^k$. The poor performance of $\bar{q}_k(y)$ is because the error control will be valid only for the function value and there is no control on its gradient.

We present the procedure to estimate $\nabla\phi(y)$ where $\phi(y) \in C^1[0,1]$ by a polynomial $q_k(y)$.

Step 1. Determine the overall error bound $\epsilon > 0$.

Step 2. Choose the degree $k \geq 10$.

y	$\phi(y)$	$\bar{q}_k(y)$	$q_k(y)$	$\nabla\phi(y)$	$\nabla\bar{q}_k(y)$	$\nabla q_k(y)$
0.0	1.000	0.996	0.992	-3.14	-3.54	-3.16
0.2	0.412	0.411	0.410	-2.54	-2.37	-2.48
0.4	0.049	0.063	0.058	-0.97	-1.01	-0.94
0.7	0.191	0.189	0.195	1.85	1.92	1.83
0.9	0.691	0.718	0.702	2.99	3.07	3.08
1.0	1.000	1.012	1.023	3.14	2.65	3.28

Table 2: A comparison of the approximations of the function $1 - \sin \pi y, y \in [0, 1]$.

Step 3. Calculate $k + 1$ coordinates, $(y_s, \phi(y_s)), s = 1, \ldots, k + 1$ which are determined as Chebyshev nodes on $[0, 1]$.

Step 4. Solve (13) and construct $q_k(y)$.

Step 5. Use $q'_k(y)$ as an approximation of $\phi'(y)$.

3. Computational complexity

The overall computational performance is also determined by the choice of the main algorithm. In this research, we adopt the gradient mapping method in [8] as our primary algorithm for two reasons: first, this method terminates within a polynomial number of iterations, second, only the first order information such as functional value and gradient are required. We cite the results from [8] as the foundation of our complexity analysis and we refer readers to that book for more technical details.

We now present the number of arithmetic operations including additions, multiplications, divisions, and comparisons for our approach to show that our approach leads to the optimal solution at a cost of polynomial complexity. First, we show the computational complexity of evaluating $g(x)$ and $\nabla g(x)$ at a given x. Second, we present the overall complexity with a first-order method as the main algorithm. In this analysis, there are a few required parameters including the significance level α; the degree of the approximation polynomial k; the dimension of x, n; and the overall accuracy, $\epsilon > 0$. We must remark that the arithmetic operation count is a measure of computational complexity which ignores the fact that adding or multiplying large integers or high-precision floating point numbers is more demanding than adding or multiplying single-digit integers.

Proposition 1. *If the number of arithmetic operations to evaluate $g_i(x_s^i)$ is bounded by a fixed value P, to construct model* (13)*, it requires up to $(k + 1)^2 O(k) + (k + 1)P$ arithmetic operations.*

Proof: Model (13) needs $k+1$ times of $g_i(x_s^i), s = 1, \ldots, k+1$ which takes $(k+1)P$. The construction of polynomial will need to calculate $\psi_0(x^i), \ldots, \psi_k(x^i)$. Since these terms are simple polynomials and each one of their calculations only takes up to $C \times k$ arithmetic operations where C is a large constant, we thereby

consider that each term will take up to $O(k)$ arithmetic operations regardless trivial differences among them. Thus, in order to calculate $\psi_0(x^i), \ldots, \psi_k(x^i)$, which means $k+1$ times of $O(k)$ arithmetic operations for each item $s, s = 1, \ldots k+1$. Thus, we need up to $(k+1)^2 O(k) + (k+1)P$ arithmetic operations to construct model (13). □

Model (13) has $k+1$ variables and the number of arithmetic operations required is $O[(k+1)^3]$. Thus, we need

$$n\{O[(k+1)^3] + (k+1)^2 O(k) + (k+1)P\}$$

arithmetic operations to obtain the *approximates* of $g(x)$ and $\nabla g(x)$. Since the calculation of derivatives of the polynomial is "transparent", the number of operations will be trivial and not cause any computational concerns.

By adopting the gradient mapping method in [8], we have the following result,

Proposition 2. *The gradient mapping method takes at most*

$$\frac{1}{\ln[2(1-\kappa)]} \ln \frac{t_0 - t^*}{(1-\kappa)\epsilon} \tag{17}$$

iterations to obtain an $\epsilon-$optimal solution where κ is an absolute constant (for example $\kappa = 0.25$) and t_0, t^* are the progressively updated penalty coefficients.

The proof is in [8]. In the proof, both κ and $t_0 - t^*$ are well bounded values. Thus, the value of (31) will be bounded as well. In [3], the authors suggest that the number of algorithmic iterations will be as many as 30. Based on this result, we show that

Theorem 6. *The overall number of arithmetic operations towards an ϵ-optimal solution will be*

$$n\{O[(k+1)^3] + (k+1)^2 O(k) + (k+1)P\} \frac{1}{\ln[2(1-\kappa)]} \ln \frac{t_0 - t^*}{(1-\kappa)\epsilon}$$

when we use the gradient mapping algorithm.

For instance, when we have a chance-constrained optimization with $n = 10,000$ variables. We choose to approximate this constraint by polynomials with $k = 13$ degree. Suppose $\epsilon = 0.01$, $m = 100$, $\kappa = 0.25$, and $t_0 - t^* = 100$, the number of arithmetic operations should be

$$10000\{O(14^3) + 14^2 O(14) + 14P\} \frac{1}{\ln[2 \times 0.75]} \ln \frac{100}{0.75 \times 0.01}.$$

The overall number of arithmetic operations should be in the level of 1×10^{16} arithmetic operations. Modern computers are far more capable of handling such a scale calculation. Recently, Intel Corporation demonstrated a single x86-based desktop processor sustaining more than a Tera-FLOP (10^{12}). This means that solving such a large scale chance-constrained optimization problem on an average desktop computer will only take several hours. Our computational performance supports this conclusion.

4. Evaluation of the cumulative distribution function

In the previous sections on the approximation approach and the complexity, it seems that the evaluation of $F_\xi(x)$ can be completed easily. In reality, the evaluation of $F_\xi(x) := \mathbb{P}(Dx \geq \xi)$ is a great deal of challenges because it is a multivariate integration and computationally demanding. In particular, when the distribution of ξ is assumed to be log-concave without a closed-form, explicitly evaluating $F_\xi(x)$ is nearly impossible due to the multivariate integration. We thus need to adopt Monte Carlo to bypass the multivariate integration. The simulation error can be well controlled and if the result is not satisfactory, we can always increase sample size for an improvement.

Consider an independent and identically distributed (iid) sample with sample size N, $\xi^1, \ldots, \xi^N$. Let

$$F_\xi^N(x) := \frac{\sum_{i=1}^N I(\xi^i \leq Ax)}{N} \tag{18}$$

where $I(a) = 1$ when $a \succeq 0$, $a \in \mathbb{R}^s$ and $I(a) = 0$ otherwise. $F_\xi^N(x)$ refers to the estimated cumulative distribution of ξ based on the sample $\xi^1, \ldots, \xi^N$ and

$$g^N(x) := \log(1-\alpha) - \log\left(\frac{\sum_{i=1}^N I(\xi^i \leq Ax)}{N}\right).$$

The introduction of Monte Carlo will not complicate the computational complexity results. When adopting simulation to evaluate $F_\xi(x)$, we may have a different number of arithmetic operations P_N rather than P at a given sample size N. It is less of a concern as long as the P_N is well bounded and P_N is the number of algorithmic operations with respect to a sample size of N. Thus, P_N is well bounded as long as the values of P and N are bounded by the problem input. Consider

$$g_i^N(x_s^i) = \log(1-\alpha) - \log(\frac{\sum_{j=1}^N I(\xi^j \leq D[x^1; \ldots; x^{i-1}; x_s^i; x^{i+1}; \ldots; x^n])}{N})$$

where $D \in \mathbb{R}^{m \times n}$. For the indicator function, it takes $m(n+n-1)$ additions and multiplication, m comparisons against the m-dimensional ξ^j, and another m comparisons with 0. We assume that the calculated values will be properly stored in memory. We need to repeat such a calculation N times with an additional N additions and divisions to calculate the average. If we count the logarithmic calculation evenly as other arithmetic operations, we need 2 more logarithmic operations and one addition. Thus, the total number of arithmetic operations will be

$$N(m(n+n-1)+m+m) + N - 1 + 1 + 1 + 1 = N(m(2n-1)+2m) + N + 2$$

which is expected to be bounded with predetermined N and P.

The incorporation of Monte Carlo removes the computationally demanding multivariate integration. However, Monte Carlo complicates the problem that

the obtained optimal solution becomes a random variable. Thus, an ideal result will be that the obtained optimal converges to the true optimal in probability. We define

$$X := \{x | f_j(x) \leq 0, j \notin I\}.$$

and X is a compact set. The original chance constrained optimization model is

$$\min \{f_0(x) |\ g(x) \leq 0,\ x \in X\}. \tag{19}$$

The interim problem, where $g(x)$ is replaced by the sample average $g^N(x)$, is

$$\min \{f_0(x) |\ g^N(x) \leq 0,\ x \in X\} \tag{20}$$

and the model we actually solved is:

$$\min \{f_0(x) |\ Q_k(x) \leq 0,\ x \in X\} \tag{21}$$

where $Q_k(x) : \mathbb{R}^n \to \mathbb{R}$ is the polynomials of degree k at an aggregated level for all n dimensions of x of the function $g^N(x)$.

Let x^*, $\hat{x}^*$, and $\hat{x}_k^*$ denote the optimal solutions from (19), (20), and (21), respectively. These problems can be written as unconstrained optimization problems:

$$\min\ f_0(x) + tQ_k(x)_+,\ t > 0, x \in X,\ Q_k(x)_+ := \max\{0, Q_k(x)\} \tag{22}$$

$$\min\ f_0(x) + tg^N(x)_+,\ t > 0, x \in X,\ g^N(x)_+ := \max\{0, g^N(x)\} \tag{23}$$

$$\min\ f_0(x) + tg(x)_+,\ t > 0, x \in X,\ g(x)_+ := \max\{0, g(x)\} \tag{24}$$

and we have the following theorem for the penalty method.

Theorem 7. *Let there exist a value $\bar{t} > 0$ such that the sets*

$$S_1 = \{x \in X | f_0(x) + \bar{t}Q_k(x)_+ \leq f_0(\hat{x}_k^*)\}$$

$$S_2 = \{x \in X | f_0(x) + \bar{t}g^N(x)_+ \leq f_0(\hat{x}^*)\}$$

$$S_3 = \{x \in X | f_0(x) + \bar{t}g(x)_+ \leq f_0(x^*)\}$$

are bounded. Then

$$\lim_{h\to\infty} f_0(x_h) = f_0(\hat{x}_k^*),\ \lim_{h\to\infty} Q_k(x)_+ = 0,\ x_h \in S_1$$

$$\lim_{h\to\infty} f_0(x_h) = f_0(\hat{x}^*),\ \lim_{h\to\infty} g^N(x)_+ = 0,\ x_h \in S_2$$

and

$$\lim_{h\to\infty} f_0(x_h) = f_0(x^*),\ \lim_{h\to\infty} g(x)_+ = 0,\ x_h \in S_3$$

where k represents the degree of polynomial and $\{x_h\}$ is the sequence of points generated by the main algorithm.

The proof is in [8] as a general conclusion for the penalty method and the choice of $\bar{t}$ is rather theoretical and symbolic. Thus, we define

$$u_k(x) := f_0(x) + \bar{t}Q_k(x)_+,\ \bar{u}(x) = f_0(x) + \bar{t}g^N(x)_+, \text{ and } u(x) = f_0(x) + \bar{t}g(x)_+$$

Consider a sequence of functions $u_k : \mathbb{R}^n \to \mathbb{R}$. It is said that u_k epi-converges to a function $\bar{u}$ if the epigraphs of the functions u_k converge, in a certain set valued sense, to the epigraph of $\bar{u} : \mathbb{R}^n \to \mathbb{R}$. In order to establish connection between $\hat{x}_k^*$ and x^*, we need two phases. First, we need to show the convergence from $\hat{x}_k^*$ to $\hat{x}^*$ and from $\inf u_k$ to $\inf \bar{u}$. Second, we need to show the convergence at least in probability from $\hat{x}^*$ to x^*.

4.1. Convergences of $\hat{x}_k^*$ to $\hat{x}^*$ and $\inf u_k$ to $\inf \bar{u}$

In order to show the convergences of $\hat{x}_k^*$ to $\hat{x}^*$ and $\inf u_k$ to $\inf \bar{u}$, we need the following convergence in minimization theorem (see [10]):

Theorem 8 (convergence in minimization). *Suppose the sequence $\{u_k\}, i = 1, \ldots$ is eventually level-bounded and $\{u_k\}$ epi-converges to $\bar{u}$ with u_k and $\bar{u}$ lower semi-continuous and proper. Then*

$$\inf u_k \to \inf \bar{u} \tag{25}$$

while $\{u_k\}$ is indexed over a sub-sequence of $\mathbb{Z}_+$ containing all ν beyond some $\bar{\nu}$. The sets argmin u_k are nonempty and form a bounded sequence with

$$\limsup_k (\textit{argmin } u_k) \subset \textit{argmin } \bar{u}. \tag{26}$$

Indeed, for any choice $\epsilon_k \to 0$ and $\hat{x}_k^* \in \epsilon_k$-argmin u_k, the sequence $\{\hat{x}_k^*\}$ is bounded such that all of its cluster points belong to argmin $\bar{u}$. If argmin $\bar{u}$ consists of a unique point $\hat{x}^*$, one must actually have $\hat{x}_k^* \to \hat{x}^*$.

The proof is in [10] and this theorem plays a central role to establish the first connection. With a large enough N,$g^N(x)$ is proper, continuous, and level-bounded and so is $\bar{u}$ *in probability*. Our approximation approach generates a good approximation, $Q_k(x)$, to $g^N(x)$ within a uniformly controlled range of error. Therefore, functions Q_k and u_k on X should be proper, continuous, and level-bounded as well. When we use the same sample in our approximation approach, $g^N(x)$ is convex and so is $Q_k(x)$. When u_k and $\bar{u}$ are strictly convex, there will be *unique* optimal solutions $\hat{x}_k^*$ and $\hat{x}^*$ for (22) and (23), respectively.

If the sequence of functions $\{u_k\}$ epi-converges to $\{\bar{u}\}$, we can then apply theorem 8 to establish the convergences of $\hat{x}_k^*$ to $\hat{x}^*$ and $\inf u_k$ to $\inf \bar{u}$. We need to prove the uniform convergence of the sequence of functions $\{Q_k\}$ to g^N because of the following proposition:

Proposition 3. *With a large enough N, if $\{Q_k(x)\}$ epi-converges to $g^N(x)$ on X and f_0 is continuous on X, then the sequence of functions $u_k(x) := f_0(x) + \bar{t}Q_k(x)_+$ epi-converges to $\bar{u}(x) := f_0(x) + \bar{t}g^N(x)_+$ for any $\bar{t} > 0$ in probability.*

Therefore, it is adequate to show that $\{Q_k(x)\}$ epi-converges to $g^N(x)$ on X under a certain sample. We have the proposition that states the epi-convergence from uniform convergence as follows:

Proposition 4. *With a large enough N, consider $\{Q_k(x)\}$ and $g^N(x) : X \to \mathbb{R}$, if the functions $Q_k(x)$ are continuous on X and converges uniformly to $g^N(x)$ on X, then $Q_k(x)$ epi-converges to $g^N(x)$ relative to X in probability.*

The above two propositions are proved in [10, Chapter 7]. We complete this phrase by the following theorem

Theorem 9. *With a large enough N, the sequence of function $\{Q_k(x)\}$ uniformly converges to $g^N(x)$ on X in probability.*

Proof: Since $Q_k(x)$ and $g^N(x)$ are continuous on X, then $\{Q_k(x)\}$ and $g^N(x)$ are bounded. With our approximation approach at any given $\epsilon > 0$, let the sequence of function be indexed with $\bar{k} \geq \min \left\{ k : \frac{1}{2^k(k+1)!} M < \frac{\epsilon}{n} \right\}$, we have

$$|Q_k(x) - g^N(x)| \leq \epsilon \text{ for all } x \in X$$

Thus, we claim that $\{Q_k(x)\}$ uniformly converges to $g^N(x)$ *in a bounded sense.* □

From the above uniform convergence, we can immediately show that $\{Q_k(x)\}$ epi-converges to $g^N(x)$ on X by applying propositions 3 and 4.

4.2. Convergences of $\hat{x}^*$ to x^* and $\inf \bar{u}$ to $\inf u$

We must remark that $\bar{u}$ is also a sequence of functions by distinct iid samples with ascending sample size N and let $\{\bar{u}^N\}$ denote this sequence of functions indexed by the corresponding sample size. Let $\hat{x}^*_N$ denote the point which minimizes $\bar{u}^N$ on X. By the law of large numbers, for a $x \in X$,

$$\lim_{N\to\infty} \mathbb{P}(|\bar{u}^N(x) - u(x)| > \epsilon) = 0, \text{ or equivalently writing as } \bar{u}^N(x) \to_p u(x) \text{ as } N \to \infty \tag{27}$$

which is called *pointwise convergence in probability of random convex functions.*

Theorem 10. *If the sequence of functions $\{\bar{u}^N\}$, at $x \in X$, $\bar{u}^N(x) \to_p u(x)$ as $N \to \infty$ and $\bar{u}^N$, u are convex on X, then*

$$\sup_{x\in X} |\bar{u}^N(x) - u(x)| \to_p 0 \text{ as } n \to \infty$$

Proof: Let $x_1, x_2, \ldots$ be a countable dense set of points in X. Since $\bar{u}^N(x_1) \to_p u(x_1)$ as $N \to \infty$, there exists a sub-sequence along which convergence holds almost surely. Along this sub-sequence, $\bar{u}^N(x_2) \to_p u(x_2)$ so a further sub-sub-sequence exists along which also $\bar{u}^N(x_2) \to u(x_2)$ almost surely. Repeating this argument by applying cumulative sub-sequences k times, we have $\bar{u}^N(x_j) \to u(x_j)$ almost surely for $j = 1, \ldots, k$. Now consider the new sub-sequence formed by taking the first element of the first sequence, the second element of the second,

and so on. Along the new sub-sequence we must have $\bar{u}^N(x_j) \to u(x_j)$ almost surely. Thus, we have this corollary:

$$\sup_{x \in X} |\bar{u}^N(x) - u(x)| \to 0 \text{ almost surely along this sub-sequence.}$$

Therefore, for any sequence, a further sub-sequence can be constructed along which $\sup_{x \in X} |\bar{u}^N(x) - u(x)| \to 0$ almost surely which automatically implies

$$\sup_{x \in X} |\bar{u}^N(x) - u(x)| \to_p 0 \text{ along the whole sequence.}$$

□

As an immediate result, we have

Corollary 1. *Suppose u has a unique minimum at $x^* \in X$. Let $\hat{x}_N^*$ minimize $\bar{u}^N$, then $\hat{x}_N^* \to_p x^*$ as $n \to \infty$.*

This proof is a simple $\epsilon - \delta$ argument.

Hence, we conclude that the sequence of the solution of (21) uniformly converges the optimal solution of (19) in probability.

5. Implementation and sample results

We code our approach to solve model (CCCFL). Software implementation includes five components: the main code, the function to calculate the functional value of $g(x)$, the function to calculate the gradient of $g(x)$, $\nabla g(x)$, the function to calculate the Chebshev nodes, and the software utility to load the model into the compatible format of certain software packages. In our numerical experiments, we use the platform of Matlab R2013a with the third-party Disciplined Convex Programming developed by Stanford University. The hardware is a HP workstation running Debian Linux Wheezy with 16G DDR3 memory and Intel i7-4770 CPU. The computational architecture is x86-64 and so do both Matlab and CVX package.

Since the chance constrained optimization problem is commonly used for business planning problems, we design the interface of modeling to be business friendly that we use Microsoft Excel templates to collect and organize the modeling parameters. All the parameters such as A, D, b, c, α are worksheets which will be loaded by the software utility. We illustrate the structure of our package in Figure 1. The main code will briefly check the dimensions of the variables and exit if there is a mismatch. If the model is successfully accepted by the main code, you may see the algorithm running until it terminates at the solution which satisfies the stopping criterion. We use this package to solve two chance constrained optimization problems in the field of financial planning, and transportation.

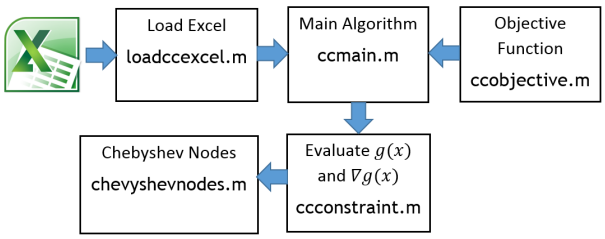

Figure 1: Functions of the software package

5.1. Cash matching

We tested our approach along with software packages on the chance-constrained optimization problem named "cash matching". This example first appears in [5] and it has been repeatedly used in many papers and talks. The pension fund of a company has to make payments for the next 15 years. Payments shall be covered by investing an initial capital $K = \$250,000$ in bonds of three different types by monetary amounts x_1, x_2, x_3. The objective is to maximize the final amount of cash after 15 years, subject to the constraints of covering payments in all years. Let $\alpha_{ij}, i = 1, 2, 3, j = 1, \ldots, 15$ denote the earnings of i^{th} bonds at year j. The costs of bonds are $[\gamma_1\ \gamma_2\ \gamma_3]' = [980\ 970\ 1050]'$ respectively. β_j is the payment by years.

The cash available at the end of year j is

$$K - \sum_{i=1}^{3} \gamma_i x_i + \sum_{k=1}^{j} \sum_{i=1}^{3} \alpha_{ik} x_k - \sum_{k=1}^{j} \beta_k \geq 0$$

The yearly payments are random vectors that happen to be component-wise independent, which is a coincidence. In fact, our approach would be able to solve the cash matching problem with non-independent random vectors as well. In order to ensure the timely payment, we need to impose the chance constraint on (28), i.e.,

$$\mathbb{P}\{K - \sum_{i=1}^{3} \gamma_i x_i + \sum_{k=1}^{j} \sum_{i=1}^{3} \alpha_{ik} x_k - \sum_{k=1}^{j} \beta_k \geq 0, j = 1, \ldots, 15\} \geq 1 - \alpha$$

The random vector in (28) is $[\beta_1; \ldots; \sum_{k=1}^{15} \beta_k]$ and $\mathbb{P}$ represents the probabilistic measure for the joint distribution of the random vector.

The stopping criterion is to find an ϵ-optimal where $\epsilon = 0.01$. We use the starting point at $[70, 80, 85]$, service level at 0.96, and step size at $3/k$. The starting point is determined by relaxing the chance constraint with multiple linear constraints. The random variables are replaced by their mean values. The once-difficult problem thus becomes a linear programming which can be solved easily. The obtained solution from the linear programming will become the starting solution. The algorithm terminates at the solution $[69.63, 86.17, 80.72]$

after 13 iterations. When we can explicitly calculate the functional value of the chance constraints, the optimal solution will be optimal to the original. By choosing a different starting point, the computational cost may vary, and when the starting point is "distant" from the optimal solution, we need a different number of iterations because the gradient mapping method will surely converge to the optimal regardless the starting point.

5.2. Stochastic multi-commodity network flow

This problem is introduced in [2]. In this example, let us consider a stochastic multi-commodity network flow problem with the node set $\mathcal{V}$ and arc set $\mathcal{A} \subset \mathcal{V} \times \mathcal{V}$. For each pair of nodes $(k, l) \in \mathcal{V} \times \mathcal{V}$, there is a random quantity d_{kl} to be shipped from k to l. The objective is to find arc capacities $x(a), a \in \mathcal{A}$, such that the network can carry the flows with a sufficiently large probability $1 - \alpha$ and the capacity expansion cost $c'x$ is minimized. The network structure is shown in Figure 2, and the cost factor is in Table 3. The demand is symmetric,

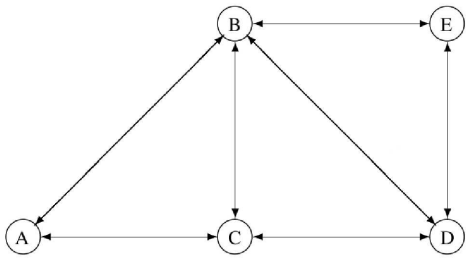

Figure 2: The graph of the stochastic multi-commodity network flow

From	A	A	B	B	B	C	D
To	B	C	C	D	E	D	E
Unit cost	\$310	\$230	\$250	\$180	\$350	\$400	\$270

Table 3: Unit costs by network arcs

i.e., $d_{kl} = d_{lk}$ for all the arcs and

$$d_{kl} = 0.1D + \xi_{kl}, \text{ where } D \sim \mathbb{N}(30, 5^2),\ \xi_{kl} \sim \mathbb{N}(0,\ 0.25^2)$$

The model in the original paper is a two stage stochastic programming and we need to remove the second stage variables y (notation in [2]) and impose chance constraint on the *equivalently* re-written affine inequalities.

In our model, there are two types of inequalities: inequalities included in chance constraint; and inequalities as constraints formulating the feasible set. We still use $x(a), a \in \mathcal{A}$ as the decision variable. For individual nodes, we need the incoming capacity to match the outgoing capacity. For example, node A has.

$$x_{CA} + x_{BA} = x_{AB} + x_{AC} \tag{28}$$

where the left-hand side is the total volume of incoming capacity and the right-hand side is the total volume of outgoing capacity. The notation x_{AB} denotes the capacity for the traffic *from A to B*. In addition, we need to consolidate multiple nodes as a new virtual node to avoid a potentially isolated network. We impose the equality constraint on the virtual nodes of $A\&B$ as the follows.

$$x_{EB} + x_{DB} + x_{CB} + x_{CA} = x_{BC} + x_{AC} + x_{BE} + x_{BD} \tag{29}$$

Similarly, the left-hand side is the total incoming capacity to the virtual node $A\&B$, and the right-hand side is the total outgoing capacity. There is no need to impose similar equality constraint for the consolidation of three or more nodes because we only have 5 nodes in total. Constraints (28) and (29) define the feasible set X and they will not appear in the chance constraint.

Another type of inequalities will be incorporated into the chance constraint. For individual node, we need to ensure that the random demand will be less than the capacity by large chance, i.e., $1 - \alpha$. For example, node A imposes

$$x_{AB} + x_{AC} \geq d_{BA} + d_{CA} + d_{DA} + d_{EA} \tag{30}$$

For the virtual node of $A\&B$, consolidation of nodes A and B, the inequality becomes

$$x_{BC} + x_{AC} + x_{BE} + x_{BD} \geq d_{AC} + d_{AD} + d_{AE} + d_{BC} + d_{BD} + d_{BE} \tag{31}$$

Chance constraint includes constraint type of (30) and (31). We need to consolidate demands as well. For example, in constraint (30), the right-hand side is the total demand from node A. We consolidate random variables d_{BA}, d_{CA}, d_{DA}, d_{EA} into one random variable. Thus, the variance-covariance matrix should be calculated accordingly.

From-To	Exact solution	SS=200	SS=500	SS=1,000	SS=10,000	SS=100,000
A-B	10.93	10.86	10.99	10.94	10.94	10.90
A-C	3.75	3.89	3.77	3.76	3.76	3.73
B-C	7.30	7.11	7.33	7.30	7.30	7.27
B-D	7.47	7.22	7.51	7.47	7.47	7.46
B-E	10.83	10.53	10.90	10.85	10.85	10.81
C-D	3.63	3.97	3.65	3.63	3.63	3.60
D-E	3.23	3.35	3.27	3.25	3.25	3.25
Time	1380s	118s	123s	140s	336s	2252s
Objective	26,981	27,274	27,149	27,031	27,031	26,859
Iterations	37	19	19	20	20	20
$\mathbb{P}(Ax \geq \xi)$	0.9500	0.9517	0.9490	0.9412	0.9413	0.9480

Table 4: Exact and simulation-based optimal solutions by our proposed approach, SS represents the term "sample size" with starting point $x = [12; 12; 4; 4; 8; 8; 8; 8; 12; 12; 4; 4; 4; 4]$.

In this example, we need to show the stability of our approach under the complication of Monte Carlo. We first present the optimal solution obtained

from the exact evaluation of the joint normal distribution as the benchmark. We then replace the exact calculation of the joint normal distribution by its sample average. We demonstrate the performances of our approach under sample sizes of 200, 500, 1,000, 10,000, and 100,000 to suggest that our method will lead to stable numerical results. Surprisingly, we find that Monte Carlo leads to considerable saving in terms of calculation time; the calculation of the joint normal distribution function could be time consuming and we can only calculate the cumulative distribution function value of ξ with a limited number of dimension.

6. Concluding remarks

This paper introduced a numerical approach for the commonly encountered chance constraint optimization problem in which the chance constraint is imposed on multiple affine inequalities with a random vector in the right-hand side. In order to preserve the convexity for both the feasible set and the chance constraint, the joint distribution of the random vector is assumed to be continuously log-concave. Under these assumptions, the problem is equivalent to a convex optimization problem. The primary challenge to solve such a problem is to efficiently evaluate the joint cumulative distribution function to calculate the function value and gradient. An explicit solution seems impossible because of the notoriously slow multivariate integration. We bypass the multivariate integration by adopting the sample average. We then developed a Bernstein polynomial-based approximation to obtain the functional value and the gradient of the chance constraint.

Our approach controls the error of function value and the error of gradient both *simultaneously* and *uniformly*. With the gradient mapping algorithm, the overall computational complexity will be polynomial and the optimal solution, although indeed a random variable, will converge to the true optimal in probability. We implement our approach for three business analytic examples: the financial planning, and supply chain management. The performances suggest that our approach yields stable solution for various scale problems. We also need to remark that the efficient implementation written in languages, especially suited to scientific computing, such as Fortran or C/C++, will considerably impact the overall performance. With the mainframe computational facilities, this specific type of chance constrained optimization should be solved within a timely manner.

[1] N. I. Achieser, *Theory of approximation*, Courier Corporation, 2013.

[2] P. Beraldi and A. Ruszczyński, *A branch and bound method for stochastic integer problems under probabilistic constraints*, Optimization Methods and Software, 17 (2002), pp. 359–382.

[3] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge university press, 2009.

[4] E. Cheney, *Introduction to approximation theory. 1966*, Chelsea, New York.

[5] D. Dentcheva, B. Lai, and A. Ruszczyński, *Dual methods for probabilistic optimization problems**, Mathematical Methods of Operations Research, 60 (2004), pp. 331–346.

[6] D. Dentcheva, A. Prékopa, and A. Ruszczynski, *Concavity and efficient points of discrete distributions in probabilistic programming*, Mathematical Programming, 89 (2000), pp. 55–77.

[7] D. Jackson, *The theory of approximation*, New York, 19 (1930), p. 30.

[8] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer, 2004.

[9] G. M. Phillips, *Interpolation and approximation by polynomials*, vol. 14, Springer, 2003.

[10] R. T. Rockafellar, R. J.-B. Wets, and M. Wets, *Variational analysis*, vol. 317, Springer, 1998.

[11] A. Shapiro and P. R. Andrzej, *Stochastic programming*, Elsevier, 2003.

[12] A. Shapiro, D. Dentcheva, et al., *Lectures on stochastic programming: modeling and theory*, vol. 9, SIAM, 2009.

[13] G. Stewart and D. R. Kincaid, *Afternotes on numerical analysis*, SIAM Review, 39 (1997), p. 153.